Vladimir Patryshev
277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

United States Patent and Trademark Office

Application No.: 09/682,576

Request for Reconsideration of Office Action regarding application 09/682,576

Dated this 20th day of September, 2004

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555
Vladimir Patryshev

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Hereby I request an interview with examiners regarding the office action on application number 09/682,586.

My preferred dates for interview are:

09/25, 09/30, 10/05, 10/07, 10/12, between 9AM and 12AM.

Vladimir Patryshev

September 20, 2004

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Vladimir Patryshev
277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

United States Patent and Trademark Office

Application No.: 09/682,576

Request for Reconsideration of Office Action regarding application 09/682,576

Dated this 20th day of September, 2004

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555
Vladimir Patryshev

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Thank you for sending me the review of my patent application.

Unfortunately, I have to request to reconsider Your Office Action regarding my application, since I disagree with the points in the rejection. The rejection mentions several publications: Tuukkanen, Berstis, and Horn.

My main argument is that, while Janne Tuukkanen's work relates to data encryption (not encoding), Berstis' patent relates to url correction, and Horn's application relates to using various encodings to transfer and represent business data, my application relates to detecting of the encoding that the user chooses on their browser, and possible transferring of this information to the server.

Below I will present some basic definitions regarding the topic of the invention, then my arguments, and, finally, a comparison table supporting my statements.

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

1. Definitions

Character Encoding

Definition A (http://en.wikipedia.org/wiki/Character_encoding)

A **character encoding** is a code that pairs a set of natural language characters (such as an alphabet or syllabary) with a set of something else, such as numbers or electrical pulses. Common examples include Morse code, which encodes letters of the Latin alphabet as series of long and short depressions of a telegraph key; and ASCII, which encodes letters, numerals, and other symbols as both integers and 7-bit binary versions of those integers.

Definition B (www.adobe.com/type/topics/glossary.html)

Character encoding is a table in a font or a computer operating system that maps character codes to glyphs in a font. Most operating systems today represent character codes with an 8-bit unit of data known as a byte. Thus, character encoding tables today are restricted to at most 256 character codes. Not all operating system manufacturers use the same character encoding. For example, the Macintosh platform uses the

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

standard Macintosh character set as defined by Apple Computer, Inc., while the Windows operating system uses another encoding entirely, as defined by Microsoft. Fortunately, standard Type 1 fonts contain all the glyphs needed for both these encodings, so they work correctly not only with these two systems, but others as well. Also see character, glyph, keyboard layout.

Data Encryption

Definition A (www.stallion.com/html/support/glossary.html)

Prevents any non-authorized party from reading or changing data. The level of protection provided by encryption is determined by an encryption algorithm. In a brute-force attack, the strength is measured by the number of possible keys and the key size. For example, a Triple-Data Encryption Standard system (3 DES) uses 112-bit or 168-bit keys and, based on currently available processing power, is virtually immune to brute-force attacks. Business to Business VPNs (Extranets) share sensitive data with multiple organizations, so demand the highest level of security. This requires public key encryption and/or secure key exchange, both of which are designed to eliminate the risk of the key becoming known to an unauthorized party.

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Definition B (http://en.wikipedia.org/wiki/Encryption)

Encryption is the process of obscuring information to make it unreadable without special knowledge. This is usually done for secrecy, and typically for confidential communications. Encryption can ensure a good measure of confidentiality (secrecy), but may not be able to provide authentication. Even when encrypted, messages can be subject to traffic analysis.

2 Discussion

2.1. ***Jaanne Tuukkanen, "Simple Script Security (SSS) v0.85beta", http://projannet.port5.com***

This article discusses the idea and the implementation of data encryption via Javascript. The purpose of this is to ensure that data are transferred in obscured form over the network, while keeping things simple on the client side, and, since the implementation is in Javascript, it does not depend on the browser implementation, and can be used with virtually every browser, even on mobile devices (those of them that support Javascript). The article has nothing to do with

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

international characters, character encoding, and text parsing or display.

2.2. ***Berstis , et al., U.S. patent 6,092,100***

This patent solves a problem of URL resolving when the URL may not be well-formed, or may be entered only partially. The patent offers a heuristics to use to suggest the user that is typing a URL in the address field in Navigation Toolbar of a browser. This technique is now ubiquitous: in many browsers, if you type an address, the browser opens a drop-down list with suggestions - based, probably, on this invention. This patent has nothing to do with the page content or the encoding used for representing characters of the page, or with client-server communication.

2.3. ***Horn et al, UPSTO publication 2003/0156688 A1***

This publication is dedicated to a global, i.e. multiligual, electronic commerce system. The data in the system may be in different alphabets, thus belonging to different character sets, and, hence, represented in the system using various character encodings.

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Most, if not all, databases now support various character sets; some of databases use their internal encoding for all the character sets supported, while others can be tuned to used a specific encoding for a specific language.

Most browsers also support a multiplicity of character setcs, and render pages in different encoding. The browser chooses the encoding based on 'charset' metatag in HTTP, on its proprietary algorithm of detecting the character set, and on the user's choice: for instance, in Netscape, via View->Character Coding, one may swich between different encodings, and the browser will use that encoding while interpreting and rendering the contents of a web page or of a file.

Horn et al. use in their publication these two features to store multilingual data in their database and to send them over to the client's browser, so that texts in many languages can be seen. Since the system they patent already knows which language and which encoding is used for a particular set of data, there is no need to try to detect the encoding that the user has currently chosen. The system assumes that the encoding is always what was specified when the data are sent over from the server to the browser (which, as practice shows, is not always true: the user can switch current encoding for some reason - and

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

then the data sent back to the server will look "corrupt", since the server assumes the same encoding that it sent the data in). The wrong encoding on the client side may be the result of the browser setting the encoding wrongly. This actually happens with some versions of Internet Explorer.

2.4. My application, 09/682,576, does not deal with the data on the server side, and does not specify which encoding to use for which data, or whether to use it. It solves a specific problem of detecting, dynamically, the encoding that the browser currently has, whether set from the metainformation on the web page, from the browser's own guess, or manually by the user. The information obtained using my method can be a) used for manipulating data using scripting languages on the client side, and/or sent back to the server, together with the text entered by the user, and to be later used to decode the user input properly. Unfortunately, so far none of the browsers, in their object model, provides this information directly. The application deals with this problem, and successfully resolves it.

3. Regarding Claim Rejections

3.1. **Regarding claim 1:**

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Berstis does not mention "encoding" anywhere in the patent text. Horn in [102]-[105] suggests to use Unicode for data encoding, and writes, in [105], that some systems convert data from Unicode to client-requested encoding.

My claim 1 does not recommend any specific encoding, but offers a method for detecting such encoding. Neither Berstis, nor Horn ever mentions encoding detection.

Berstis' detection engine detects the possible candidate URLs, not the character encoding. Berstis' method is fuzzy, and eventually involves the user participation, and it cannot be applied to fully-automated encoding detection as described in my claim.

More, Unicode cannot be used for Berstis' case, because Unicode is not used for encoding multilingual characters in URL strings. URL has its own specific standard for such characters.

My claim also does not apply to converting between character sets. It is the function of the software on the server side to determine,

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

whether to convert the data, once the data are received and the character set is known, or keep it, together with the character set.

3.2. Regarding claim 2:

Berstis' invention describes the behavior of the browser, and it cannot be applied to the scripts which in the majority of browsers, for the sake of security, cannot retrieve the information from address text input, and thus cannot access the URL the user types. Neither [180] nor [105] in Horn mention any scripting. In other places, Horn mentions JavaScript as a took that could be used for some purposes, together with other tools; in [0237] Jscript is mentioned as one of the existing scripting languages. [0293] again repeats the common knowledge that JavaScript and Jscript uses ASCII, but can contain non-ASCII Unicode characters in string literals and comments.

3.3. Regarding claim 3:

Horn mentions Utf-8 and multi-byte encoding as one of the existing encodings. This is a common knowledge; the full list of encodings supported in the Internet can be found at www.iana.org/assignments/character-sets. My application is aware of

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

this multiplicity of encodings, and is dedicated to finding out which one is currently used. Claim 3 consists of detailing the method of encoding detection: start with Utf, then multi-byte, then single-byte encodings.

3.4. Regarding claim 4:

Horn does not teach "accompanying the form data sent from the web client"; this is actually a quote from my claim 4.

In [202] and [203] Horn repeats the common knowledge that form data are sent from the client to the server, and <FORM> tag is used for setting up the form in HTML. The statement in Horn [203] about "FORM request" is factually wrong, there is no FORM request in HTTP (see http://www.w3.org/Protocols/ for reference). Secure web pages do not relate to my claims.

Since Bertsis does not claim any character set or character encoding detection, it is impossible to apply Bertsis' algorithm of URL string prediction for detecting encoding, even less for conversion.

3.5. Regarding claim 5:

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Again, "correct form data conversion on the server side .. collected" is a quote from my claim 5. Horn just mentions that form data are sent from the browser to the server in HTTP requests, which is a common knowledge.

Making secure web pages are not related to my invention.

My claim is that the information collected using claims 1-4 can be then successfully used on the server to decode the bytes received from the client browser.

Here is the problem. A server send a page with a form. The server has no clue which language the user is going to use, and, when the browser sends back the bytes that represent the characters the user typed, the server cannot find out how to map the bytes to Unicode. Unless the method of my invention is used - this method provides this kind of information. No other method does.

But of course this invention assumes using HTTP for communication, HTML for data representation, <FORM> tag for writing forms in HTML, and JavaScript or some other scripting language for implementing the method. It relies on the browser's ability to represent text in the encoding chosen by the user, and resolves the problem that this information, which encoding is chosen, is, generally speaking, hidden

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

from the server and from the Object Model that a scripting language can use.

4. Comparison Chart

Please find the comparison chart on the next page.

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Feature	Patryshev, app. 09/682,576	Horn et al., US-2002/0156688	Berstis US-6,092,100	Tuukkanen,projennet.port5.com
Purpose	Detecting of browser encoding on the client side, browser and content-independent	Transferring multilingual data between database, web server and a client browser	Heuristic detection of URL while the user is typing it or when the user mistyped it, including the case of unknown, probably international, characters	Browser/server data exchange security, using known encryption algorithms
Multilingual character encodings involved	Yes	Yes	No; URL strings use its own, specific, method of encoding characters outside ASCII character set	No
Contents to which character encoding applies	Web page displayed by the browser and contents of form fields	Neb page displayed by the browser	None	N/A

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555

Detects character encoding	Yes	No	No: URL strings are required to use the URL encoding only, there is nothing to detect	No
Specifies which character encoding to use with data	No	Yes	No (there is one standard character encoding for URL strings)	N/A
Uses scripting language to solve the problem	Yes	N/A	No	Yes
Brings 100% correct, definite result	Yes	N/A	No: uses fuzzy logic, presents a user a list of possible choices	N/A

277 Esteban Way,
San Jose, CA 95119
(408) 226 1555